Technical Report: Conditional Reactive Simulatability*

Michael Backes[1], Markus Dürmuth[1], Dennis Hofheinz[2], and Ralf Küsters[3]

[1] Saarland University, Saarbrücken, Germany, {backes|duermuth}@cs.uni-sb.de
[2] CWI, Cryptology and Information Security Group, Prof. Dr. R. Cramer, Dennis.Hofheinz@cwi.nl
[3] ETH Zürich, Switzerland, ralf.kuesters@inf.ethz.ch

November 2, 2008

Abstract. Simulatability has established itself as a salient notion for defining and proving the security of cryptographic protocols since it entails strong security and compositionality guarantees, which are achieved by universally quantifying over all environmental behaviors of the analyzed protocol. As a consequence, however, protocols that are secure except for certain environmental behaviors are not simulatable, even if these behaviors are efficiently identifiable and thus can be prevented by the surrounding protocol.

We propose a relaxation of simulatability by conditioning the permitted environmental behaviors, i.e., simulation is only required for environmental behaviors that fulfill explicitly stated constraints. This yields a more fine-grained security definition that is achievable for several protocols for which unconditional simulatability is too strict a notion, or at lower cost for the underlying cryptographic primitives. Although imposing restrictions on the environment destroys unconditional composability in general, we show that the composition of a large class of conditionally simulatable protocols yields protocols that are again simulatable under suitable conditions. This even holds for the case of cyclic assume-guarantee conditions where protocols only guarantee suitable behavior if they themselves are offered certain guarantees. Furthermore, composing several commonly investigated protocol classes with conditionally simulatable subprotocols yields protocols that are again simulatable in the standard, unconditional sense.

1 Introduction

Simulatability-based Security. As a tool to define and prove the security of cryptographic protocols, the concept of simulatability has a long history, e.g., [2–6]. In recent years, in particular the general simulatability frameworks of reactive simulatability [7–9] and universal composability [10, 11] proved useful for analyzing security properties of cryptographic protocols in distributed systems.

One advantage of simulatability-based approaches is the simple and straightforward definition of security. Namely, security is defined by comparison to an ideal specification of the respective protocol task. Usually, such an ideal specification is given by a single machine called trusted host, which is immune to any adversarial attacks by construction. Now a

* An earlier version of this paper appeared in Proc. of *11th European Symposium on Research in Computer Security (ESORICS)* [1].

protocol is said to be secure if all of its weaknesses are already reflected in the ideal specification. More specifically, for any possible attack on the real protocol, there should be a corresponding (by construction harmless) ideal attack on the trusted host. We require that these attacks must be indistinguishable in the sense that no protocol environment can distinguish between running with the real protocol and the real attack, and running with the trusted host and the ideal attack. In that sense, the real protocol is at least as secure as the ideal specification. Because the ideal attack is to give the impression of a real attack, the ideal attacker is also called simulator.

Composition. Another advantage of such a simulatability-based definition of security is the possibility to compose protocols without loss of security. Very general composition theorems have been proven in [12, 10, 13, 14] for simulatability-based frameworks. In a nutshell, this means that any protocol M that is (in the above sense) at least as secure as an ideal specification M' can be substituted in *any* protocol context for M'. The resulting protocol that uses M will be at least as secure as the one that uses M'. On a technical level, this is not at all surprising: one could view the larger protocol simply as part of the protocol environment of M, resp. M'. Then security of M in presence of all protocol environments in particular implies security in presence of the larger protocol. However, although not surprising, this compositionality greatly aids modular protocol design: large protocols can be designed and analyzed using ideal building blocks. In a second step, these ideal building blocks can be substituted with cryptographic implementations.

This methodology is illustrated in Figure 1. In this figure, L represents a larger protocol

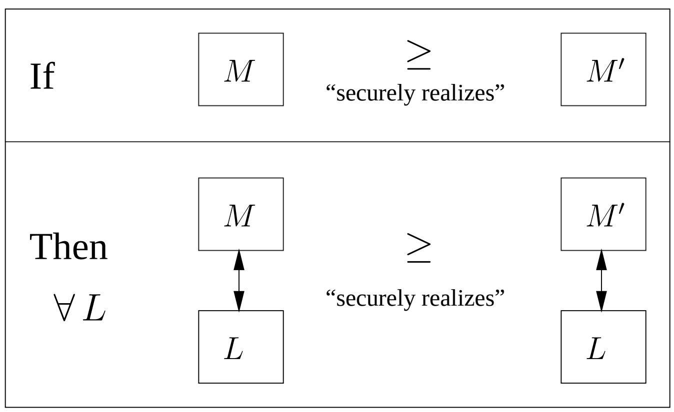

Fig. 1. Illustration of a secure composition of systems. Think of L as a larger protocol that uses M' as a(n ideal) subprotocol. Secure composition means that M' can be substituted with M if M is a secure realization of M'. In particular, L can be analyzed in combination with the ideal, easier-to-handle protocol M', while only later replacing M' with M.

that can be analyzed in combination with idealized subprotocols M' (e.g., M' could be a secure channel or an idealized signature scheme). Later, M' can be replaced with a secure instantiation (like a concrete cryptographic encryption or signature scheme) without loss of security.

As an example of the usefulness of this paradigm, general protocol constructions like the secure multi-party computation protocol of [4] can be analyzed conveniently and modularly in a simulatability-based setting [15]. Also, compositional properties are a key ingredient for the BPW model [8, 16–18] that relates security properties of abstract, Dolev-Yao style protocols with those of cryptographic implementations. This in particular enabled the first composable yet cryptographically sound security proofs of various security protocols [19–24]. Interestingly, it has even been shown that a variety of security properties are preserved under this paradigm [25–27, 25, 28–30].

The Price of Composability and the Commitment Problem. Unfortunately, such nice compositional properties are bought at a certain price. To provide an easy example, consider the task of a secure message transmission from Alice to Bob, where both already possess a common secret key for a symmetric encryption scheme. In a real protocol, Alice simply encrypts her message and sends the ciphertext to Bob. In the ideal setting, Alice simply inputs her message (unobserved by an ideal adversary) into the trusted host, who then delivers the unaltered message secretly to Bob. To show the real protocol secure, any real attack must have an ideal counterpart, such that both are indistinguishable in any protocol environment. Now observe that in the real protocol, Alice essentially commits herself to the message as soon as she sends the ciphertext to Bob. (Especially if the underlying message is sufficiently long and has enough entropy, a ciphertext already uniquely determines key and message.) Imagine a real adversary that eavesdrops Alice's ciphertext and announces it to the protocol environment, "just for the record." After this, the adversary corrupts Alice and can then, using Alice's internal state, explain to the protocol environment the observed ciphertext as an encryption of the transmitted message under the predistributed key.

But as senseless and meaningless as such an attack seems, it has no ideal counterpart. To be indistinguishable from the real attack just described, an ideal adversary has to first announce a ciphertext and only then may corrupt Alice (who now merely handed the message as input to the trusted host) to obtain her message. However, this ideal adversary already commits itself to a message when announcing the ciphertext, and it cannot explain this ciphertext as an encryption of Alice's message.

This problem is sometimes called the *commitment problem* of symmetric encryption, and it caused a surprising technical restriction in the symmetric encryption primitive in the aforementioned BPW model [8, 16, 18]. Essentially, this restriction forbids the corruption of protocol parties that have already used their secret encryption key. (Corrupting parties that did *not* use their secret key so far is fine.) This way, it is guaranteed that an ideal adversary is never forced to explain a ciphertext it made up as an encryption of a particular, a priori unknown message.

We stress however, that without such restrictions, there is no symmetric encryption scheme that could be (again, in the sense of simulatable security) at least as secure as an idealized, symbolic symmetric encryption scheme [16, 18]. This impossibility only vanishes if one accepts certain restrictions.

Another Impossibility Result: Key Cycles. As another example, consider a symbolic encryption (symmetric or asymmetric) that allows to encrypt secret keys of the scheme itself.

Such a technique can be used, e.g., for distributing an updated common secret key, or for more sophisticated authentication schemes [31]. Now as long as no *key cycles* of the form $E_{K_1}(K_2), E_{K_2}(K_3), \ldots, E_{K_{n-1}}(K_n), E_{K_n}(K_1)$ appear, standard cryptographic security notions, such as indistinguishability of ciphertexts under a chosen-ciphertext attack (IND-CCA), are sufficient to prove security of a cryptographic implementation. However, in the presence of key cycles, no standard reduction on, e.g., IND-CCA security works. This is no accident, as IND-CCA security does *not* imply security in the presence of key-dependent messages [32, 18]. In fact, it seems very hard to come up with cryptographic encryption schemes that are provably secure even in face of key cycles. Currently, only solutions in the random oracle model (a harsh abstraction from reality) are known [31, 32].

In other words, again security is only possible when certain conditions are met (namely, that no key cycles appear).

More examples. There are a number of additional examples that illustrate the demanding nature of particularly simulatable security. Very related to the commitment problem is the impossibility of a simulatably secure protocol for the cryptographic task of bit commitment [33]. Also, other important cryptographic tasks like zero-knowledge proof systems and oblivious transfer [34], as well as authenticated Byzantine agreement [35] are shown (at least unconditionally) not achievable with respect to simulatable security. The same holds for whole classes of tasks (or, functionalities) that themselves fulfil certain game-based definitions [36]. In addition, also low-level tasks such as symbolic hash functions [37] or symbolic XOR [38] are not (unconditionally) achievable.

Our Contribution: Conditional Simulatability. In this work, we propose a way to relax the demanding simulatability definition without sacrificing its nice composability properties. In a nutshell, we refine simulatability by restricting the class of allowed protocol environments (in face of which real and ideal attack must be indistinguishable). More precisely, for a real protocol M to be as secure as an ideal specification M', we demand that for every real attack on M, there is an ideal attack on M', such that no protocol environment *that fulfils a condition* π can distinguish between running with the real protocol and the real attack, and running with the trusted host and the ideal attack.

Note that (in contrast to other approaches to circumvent impossibility results, see below) we do *not* restrict the adversaries' capabilities, but *only* the considered protocol environments. The condition π we impose on the protocol environment is not fixed once and for all. Hence, in contrast to the unconditional "at least as secure as" notation, we introduce *conditional simulatability* and write that M is *at least as secure as* M' *under condition* π.

Conditional Simulatability implies Composability. When restricting our attention to protocol contexts that fulfil a certain condition π, we can of course only expect security if a larger protocol, that uses M or M', satisfies π (when considered as a protocol environment). It is immediate that this limits the compositional guarantees we obtain. However, this degradation of composability is *graceful* in the following sense: we prove that M can without loss of security be substituted for M' in larger protocols that do satisfy π. Formally, we obtain that for any larger protocol L that uses M' as a subprotocol and fulfils π, we have that the protocol "L using M" is at least as secure as "L using M'". Interestingly, this security is unconditional, since we assumed that L fulfils π unconditionally. Hence, we re-obtain full, unconditional security from conditional security under composition.

We also consider the case where the large protocol L only satisfies π if in turn some other condition τ is fulfilled. (An easy example is a protocol L for secure message transmission that uses as building block M' a trusted host for symmetric encryption. If L is never asked to transmit a certain message, it can also guarantee that it never asks M' to encrypt this message.) We prove the composition property one would expect in this situation; namely, "L using M" is at least as secure as "L using M'" *under condition* τ.

Technically, our composition theorem establishes a cryptographic statement on the acyclic composition of general assume-guarantee specifications, i.e., specifications that guarantee suitable behaviors only if they themselves are offered suitable guarantees. Assume-guarantee specifications have been well investigated in the past, mostly for non-security-specific contexts [39–42] but also specifically for security aspects [43] (but without investigations of simulatability and composition). The postulation of acyclicity applies to most cases in practice, e.g., to protocols that provide specific security guarantees to their subprotocols without making these guarantees dependent on the outputs they obtain from these subprotocols.

Interestingly, we can even prove compositionality for cyclic dependencies of such specifications, i.e., compositions of protocols that mutually promise to adhere to a certain behavior only if they mutually receive guarantees from each other. This case is technically more demanding since an inductive proof by proceeding through the acyclic dependency graph as done in the proof of the acyclic case is no longer possible. In fact, it is easy to show that for cyclic dependencies, subprotocols that are conditionally simulatable under *arbitrary* trace properties might not be securely composable. However, we prove that the theorem for the acyclic case can be carried over to the cyclic case if the constraints imposed on protocols for conditional simulatability are safety properties. Safety properties arguably constitute the most important class of properties for which conditional simulatability is used, especially since liveness properties usually cannot be achieved unless one additionally constraints the adversary to fair scheduling.

Our results are formalized in the Reactive Simulatability framework [12, 7, 9]. However, we do not use any specific characteristics of this framework, so our results can naturally be carried over to other frameworks as well, e.g., those in [11, 14].

Applying our Results. We illustrate the usefulness of our definition and the (conditional) composability guarantees that are retained by the above example of the commitment problem with symmetric encryption. We show that a secure real encryption system *does* implement a symbolic Dolev-Yao-like symmetric encryption functionality under a suitable no-commitment condition on the considered protocol environments.

We also demonstrate that in addition to circumventing known impossibility results for unconditional simulatability, the notion of conditional simulatability may also allow for securely realizing ideal functionalities at lower cost on the underlying cryptographic primitives. For instance, if Dolev-Yao style symmetric encryption permits the construction of key cycles, e.g., encrypting a key with itself, it is only securely realizable by encryption schemes that fulfill certain strong, non-standard assumptions such as the aforementioned security in presence of key-dependent messages. If, however, the functionality is conditioned to those cases that exclude key cycles, successful simulation of real attacks is possible based on weaker, more standard security notions such as IND-CCA security.

Related Work. There have been several attempts to relax simulatability to avoid impossibility results. The work closest to ours is the work on proving Dolev-Yao style symmetric encryption sound in the sense of simulatability [16]. There it was shown that Dolev-Yao style symmetric encryption can be securely realized if the environmental protocol does not cause the commitment problem and in addition key cycles are excluded. This definition thus constitutes a special case of conditional reactive simulatability yet without investigating more general conditions or corresponding compositionality aspects. Nevertheless, our work is inspired by their idea of augmenting simulatability with conditions on environments.

The impossibility of simulating attacks on bit commitment schemes was shown in [33]. The remedy proposed there was to augment the real protocol with certain “helping trusted hosts” which are, by definition, immune to any attack on the real protocol; thus, effectively this weakens the real adversary. More specifically, [33] presented simulatably secure protocols for bit commitment and zero-knowledge. However, these protocols rely on a so-called Common Reference String (CRS), which is a form of a trusted setup assumption on the protocol participants. In a similar vein, [15] shows that basically every trusted host can be realized using a CRS as a helper functionality. One point of criticism against the CRS approach is that the proposed protocols lose security in a formal and also very intuitive sense as soon as the CRS setup assumption is invalidated. The related approach [44] uses a Random Oracle (RO) instead of a CRS to help real protocols achieve simulatable security. The benefit of their construction is that the proposed protocols retain at least classical (i.e., non-simulatable) security properties when the RO assumption is invalidated. However, also there, simulatability in the original sense is lost as long as this happens.

In [45], the real and ideal adversaries are equipped with a so-called imaginary angel. This is an oracle that (selectively) solves a certain class of hard computational problems for the adversary. Under a very strong computational assumption, this notion could be shown to avoid known impossibility results for simulatability. Yet, as the imaginary angels behave in a very specific way tailored towards precisely circumventing these impossibility results, e.g., these angels make their response dependent on the set of corrupted parties, the model might be considered unintuitive. Tweaking the model to fit a specific proof technique additioally bears the danger of no longer capturing the intended properties and of complicating a validation of the model.

In [46], it is shown how to realize any trusted host in a simulatable manner, if the ideal adversary is freed from some of its computational restrictions. However, it is substantial that in their security notion, the ideal adversary is not restricted to polynomial-time, but the real adversary is. So in particular, the security notion they consider is not transitive and it is generally not easy in their framework to construct larger protocols modularly.

Outline. We first review the underlying Reactive Simulatability framework in Section 2 and subsequently define the more fine-grained version of conditional reactive simulatability in Section 3. The bulk of the paper is dedicated to the investigation of the compositionality aspects of this new security notion for both acyclic and cyclic assume-guarantee conditions (Section 4). The usefulness of conditional reactive simulatability is further exemplified in

Section 5 by showing how this notion can be exploited to cryptographically justify common idealizations of cryptography. Section 6 concludes.

2 Review of the Reactive Simulatability Framework

Our work builds upon the Reactive Simulatability framework. We will briefly review relevant definitions and refer the reader to [7] for details.

2.1 Overall Framework

A protocol is modeled as a *structure* (M, S) consisting of a set of protocol *machines* and a set of *service ports*, to which the *protocol user* connects[1]. Machines are probabilistic, polynomial-time I/O automata, and are connected by *ports*. The model differentiates in-ports and out-ports, where each out-port is connected to exactly one in-port by naming convention. Moreover, in- and out-ports may be service or non-service ports. In what follows, by S^{in} we denote the service in-ports of S and by $S^{C,out}$ the complement of M's service out-ports, i.e., the set of service in-ports of machines M connects to.

Two structures (M_1, S_1) and (M_2, S_2) are *composable* iff they connect through their respective service ports only. Their *composition* is given by $(M_1 \cup M_2, S)$ where S includes all ports from S_1 and S_2 that are not connected to another machine in $M_1 \cup M_2$.

A set of machines M is *closed* iff all ports are connected to corresponding ports of machines that are in the same set. A structure can be complemented to a closed set by a so-called *honest user* H and an *adversary* A, where H connects to service ports only, and A connects to all remaining open ports, and both machines may interact. The tuple $(M, S, \mathsf{H}, \mathsf{A})$ is then called a *configuration* of (M, S) where one of the machines H or A plays the role of the *master scheduler*, i.e., if no machine was activated by receiving a message, the master schedule is activated. A closed set C of machines constitutes a *runnable system*. The transcript of a single run is called a *trace* (often denoted by **t** and decorations thereof) and is defined to be a sequence of transitions performed by the machines. A *transition* of a machine M is of the form $(\overline{p}, s, s', \overline{p}')$ where $\overline{p}$ describes the in-ports of M along with the current message written on these ports, s is the current configuration of M, s' is a successor configuration (computed depending on $\overline{p}$ and s), and $\overline{p}'$ are the out-ports along with the output produced. We denote by $run_{C,k}$ the distribution of traces induced by runs of C with security parameter k. The restriction $\mathbf{t}\lceil_S$ of a trace **t** to a set of in-ports S is defined in the obvious way. (Note that $\mathbf{t}\lceil_S$ only depends on the first component ($\overline{p}$) of the transitions of **t**). Now, $run_{C,k}\lceil_S$ denotes the distribution of the traces induced by runs of C with security parameter k when restricted to S. The *restriction of a trace **t** to a machine* M is obtained from **t** by removing all transitions not done by M. Now, the distribution of such traces given k is denoted by $view_{C,k}(\mathsf{M})$. We refer to the k-indexed family $\{view_{C,k}(\mathsf{M})\}_k$ of these views by $view_C(\mathsf{M})$.

[1] Actually, a structure represents a protocol in a specific corruption situation. To handle different corruption situations, *systems* (i.e., sets of structures) are used. However, in the style of [7, 47], we concentrate on a given specific corruption situation for ease of presentation.

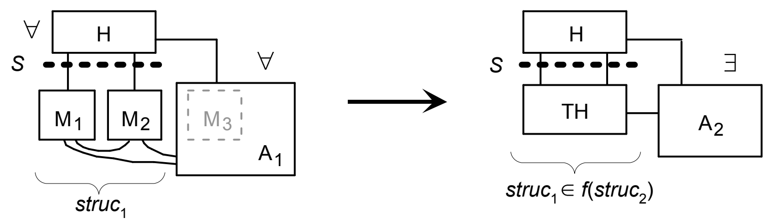

Fig. 2. Simulatability: The two views of H must be indistinguishable

2.2 Simulatability

Simulatability is used in different areas of cryptography. Informally speaking, for reactive systems it says that whatever might happen to a protocol (M, S) can also happen to another protocol (M', S). Here both protocols need to have the same set of service ports S to allow for a meaningful comparison. Typically, (M', S) is an idealization, or specification, of the protocol task that (M, S) is to implement. We therefore call (M, S) the *real* and (M', S) the *ideal protocol*. (Typically, the ideal protocol consists only of a single machine TH, a trusted host, that guarantees an ideal behaviour to a user of the protocol.) For simulatability one requires that for every configuration $(M, S, \mathsf{H}, \mathsf{A})$, with honest user H and real adversary A, there is a configuration $(M', S, \mathsf{H}, \mathsf{A}')$ of (M', S), with the same honest user H and a (possibly different) ideal adversary A', such that H cannot distinguish both scenarios. This is illustrated in Figure 2.

We define "H cannot distinguish both scenarios" in terms of computational indistinguishability: Two families $(\mathsf{var}_k)_{k\in\mathbb{N}}$, $(\mathsf{var'}_k)_{k\in\mathbb{N}}$ of random variables on common domains D_k are *computationally indistinguishable* ("$\approx$") if no polynomial-time algorithm can distinguish both distributions with non-negligible probability, i.e., if for all polynomial-time algorithms Dis the following holds:

$$\left|\mathsf{Pr}\left[\mathsf{Dis}(1^k, \mathsf{var}_k) = 1\right] - \mathsf{Pr}\left[\mathsf{Dis}(1^k, \mathsf{var}_k) = 1\right]\right| \text{ is negligible in } k,$$

where a function $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be *negligible* iff for all positive polynomials Q, $\exists k_0 \forall k \geq k_0 : g(k) \leq 1/Q(k)$.

Definition 1 (Reactive Simulatability). *Let structures (M, S) and (M', S) with identical sets of service ports be given. We write $(M, S) \geq_{\mathsf{sec}}^{\mathsf{poly}} (M', S)$, where $\geq_{\mathsf{sec}}^{\mathsf{poly}}$ is read as* computationally at least as secure as *or* securely realizes, *if for every configuration $conf = (M, S, \mathsf{H}, \mathsf{A})$, there exists a configuration $conf' = (M', S, \mathsf{H}, \mathsf{A}')$ (with the same H) such that*

$$view_{conf}(\mathsf{H}) \approx view_{conf'}(\mathsf{H}).$$

◇

One also defines *universal simulatability*, where A' in $conf'$ does not depend on H, i.e., the order of quantifiers is reversed, and *blackbox simulatability*, where A' is the composition of a fixed part Sim (the *simulator*) and A. In the sequel, we omit the superscript poly.

3 Conditional Reactive Simulatability

Reactive simulatability (Definition 1) permits configurations with arbitrary honest users H (satisfying some syntactic requirements on ports). In other words, reactive simulatability requires a faithful simulation of the combination of the real adversary and real protocol by the ideal adversary and ideal protocol for *every* honest user. This universal quantification over all honest users allows for a general composition theorem [12, 13], which says that if protocol (M, S) is as secure as protocol (M', S), then (M, S) can be substituted for (M', S) in *any* larger protocol without invalidating simulatability. For this type of compositional property, simulatability can even be shown to be necessary [48].

However, reactive simulatability may be too strict in certain practical scenarios: The simulation might fail for certain honest users, but in the application under consideration such users may not occur since the protocol in question may always be used in a certain (secure) way. For example, consider Dolev-Yao style symmetric encryption. It was shown in [16] that this kind of encryption is not securely realizable in the sense of reactive simulatability, due to the so-called commitment problem: If an encrypted message is sent to the adversary, where the adversary neither knows the message nor the key, the best the simulator can do is to create a new key and encrypt a random message with this key. If later the message becomes known, indistinguishability guarantees that the simulation is still correct. However, if later the key becomes known, the simulator has to come up with a suitable key that decrypts the chosen ciphertext to the correct message. This is not possible in general. However, in the application under consideration the way Dolev-Yao style symmetric encryption is used, e.g., by a larger protocol (representing the honest user), may guarantee that the encryption key is never exposed. It turns out that in this situation faithful simulation is still possible.

Following this idea, we propose a relaxation of reactive simulatability, called conditional reactive simulatability, where instead of quantifying over all honest users, we quantify only over those honest users which satisfy a certain condition. In this way awkward honest users which would not occur in the application anyway can be ruled out.

The conditions on honest users are expressed in terms of what we call predicates. A predicate, which is defined with respect to a set S of ports (typically service in-ports), is a set of sequences of bit strings for every port of S. Using predicates, we can restrict the kind and the order of messages on ports of S in a run of a system. To formally define these predicates, we need the following notation: For sets A and B, we denote by B^A the set of mappings from A to B. If A is a finite set, then the elements of B^A can be considered to be tuples where every component is an element of B and corresponds to an element of A. For $i \geq 0$ and a set A, we denote by A^i the set of all words over A of length i. Now, predicates are defined as follows:

Definition 2 (Predicates). *Let S be a set of ports. We call a set π with*

$$\pi \subseteq \bigcup_{i \geq 0} ((\{0,1\}^*)^S)^i$$

a predicate π over S *if the following conditions are satisfied:*

1. *If $s_1 \cdots s_i \in \pi$, $s_j \in (\{0,1\}^*)^S$, then for every $j \in \{1, \ldots, i\}$ there exists $p \in S$ such that $s_j(p) \neq \varepsilon$, i.e., for every s_j at least one port contains a non-empty message.*

2. *π is decidable in polynomial-time, i.e., there is a probabilistic polynomial-time algorithm that, on input t, outputs whether or not $t \in \pi$.*

We call $t \in \pi$ an S-trace. $\diamond$

Instead of a single predicate, one could also consider a family of predicates indexed by the security parameter. However, for the application presented in this paper, simple predicates suffice. Also, all results presented in this paper easily carry over to the case of families of predicates.

We will use the following notation. We write $\pi = \texttt{true}$ for a predicate π over S with $\pi = \bigcup_{i \geq 0}((\{0,1\}^*)^S)^i$. Furthermore, for two predicates π_1 and π_2 over two disjoint port sets S_1 and S_2, we write $\pi_1 \wedge \pi_2$ for the predicate containing all $(S_1 \cup S_2)$-traces such that for every trace in $\pi_1 \wedge \pi_2$ its restriction to S_1 and S_2 belongs to π_1 and π_2, respectively. (In a run restricted to some port set S, all entries with non-empty bit strings only on non-S ports are deleted.) Intuitively, $\pi_1 \wedge \pi_2$ represents the conjunction of π_1 and π_2.

An S-trace t' is a *prefix* of an S-trace t if there exist t'' such that $t = t' \cdot t''$ where '$\cdot$' denotes concatenation. A predicate π over S is *prefix-closed* iff for every S-trace $t \in \pi$ every prefix of t belongs to π as well. We also call such a predicate a *safety property* since once it is violated it stays violated.

Now, we say that a set of machines M fulfills a predicate π over a set of ports S, if in runs of M with any other set of machines the sequences of messages written on ports in S belong to π. More precisely, it suffices if this is true with overwhelming probability:

Definition 3 (Predicate Fulfillment). *Let M be a set of machines with service ports S and let π be a predicate over a subset S' of the ports $S^{C,out}$ of machines to which machines in M connect. Then, M* fulfills *π if for any set of machines $\overline{M}$ such that $C := \{M, \overline{M}\}$ is closed,*

$$\mathsf{Pr}_{t \leftarrow run_{C,k}}\left[(t\lceil_{S'}) \in \pi\right] \text{ is overwhelming as a function in } k.$$

$\diamond$

We are now ready to present the definition of conditional reactive simulatability.

Definition 4 (Conditional Reactive Simulatability). *Let structures (M, S) and (M', S) with identical set S of service ports be given, and let π be a predicate over a subset of the service in-ports of S. We say that (M, S)* is at least as secure as (or realizes) *(M', S)* under condition *π (written $(M, S) \geq^{\pi}_{\mathsf{sec}} (M', S)$) if for every configuration $conf = (M, S, \mathsf{H}, \mathsf{A})$ such that H fulfills π, there exists a configuration $conf' = (M', S, \mathsf{H}, \mathsf{A}')$ (with the same H) such that*

$$view_{conf}(\mathsf{H}) \approx view_{conf'}(\mathsf{H}).$$

$\diamond$

Conditional universal simulatability and *conditional blackbox simulatability* are defined with the obvious modifications.

4 Composition Under Conditional Reactive Simulatability

In this section, we present composition theorems for conditional reactive simulatability. As mentioned in the introduction, when composing protocols which assume certain conditions (predicates) to hold on their service in-ports and in turn guarantee certain conditions (predicates) to hold on service in-ports of other protocols, cyclic dependencies may occur. In what follows, we first introduce the general setting (Section 4.1) and then present general composition theorems both for the acyclic and cyclic case (Section 4.2 and 4.3). While for the acyclic case no restrictions on predicates are put, for the cyclic case we require predicates to be safety properties.

4.1 The General Setting

One would expect that a protocol M_0 (for brevity we omit the service ports) that is simulatable under condition π can be securely composed with a higher-level protocol M_1 that fulfills π. In some applications, M_1 may fulfill π only if M_1 itself is used in a certain way, i.e., a predicate, say τ, is fulfilled on the service in-ports of M_1. Then, one would expect that M_0 securely composes with M_1 as long as τ is fulfilled. More generally, we consider the composition of several protocols with assume-guarantee conditions among them. In what follows, this is formalized.

Let π and τ be predicates over S_π and S_τ, respectively, and let $\mathbf{t}$ be a trace. We say that ***t** satisfies* $\tau \to \pi$ if $\mathbf{t}\lceil_{S_\tau} \in \tau$ implies $\mathbf{t}\lceil_{S_\pi} \in \pi$.

Definition 5 (Conditional Predicate Fulfillment). *Let M be a set of machines with service ports S, τ be a predicate over a subset S_τ of S^{in}, and π be a predicate over a subset S_π of $S^{C,out}$. (Recall the definition of S^{in} and $S^{C,out}$ from Section 2.)*

Then, M fulfills π under condition τ *if $\tau \to \pi$ is satisfied with overwhelming probability no matter with which machines M interacts, i.e., for all sets $\overline{M}$ of machines such that $C := \{M, \overline{M}\}$ is closed, we have that*

$$\mathsf{Pr}_{\boldsymbol{t} \leftarrow run_{C,k}} [\boldsymbol{t} \textit{ satisfies } \tau \to \pi] \textit{ is overwhelming as a function in } k.$$

$\diamond$

In what follows, for every $i = 1, \dots, n$, let $P_i := (M_i, S_i)$ and $P_i' := (M_i', S_i)$ be real and ideal protocols, respectively. We consider the following predicates for these protocols.

Let τ_i^j be a predicate over $S_j^{C,out} \cap S_i^{in}$ (service in-ports of P_i to which P_j connects) and τ_i^{H} be a predicate over $S_i^{in} \setminus \bigcup_{j=1}^n S_j^{C,out}$ (service in-ports of P_i to which no other protocol connects). Intuitively, τ_i^j denotes the guarantees the ith protocol expects from the jth one. Analogously, τ_i^{H} specifies the guarantees the ith protocol expects from H. (Note that H may connect to all service in-ports of P_i the other protocols do not connect to.) We denote by

$$\tau_i = \tau_i^{\mathsf{H}} \wedge \bigwedge_{j \neq i} \tau_i^j \tag{1}$$

the guarantees the ith protocol expects from other protocols. Note that τ_i is a predicate over S_i^{in}.

Similarly, we now define the guarantees the ith protocol provides to other protocols. Let π_i^j be a predicate over $S_i^{C,out} \cap S_j^{in}$ (service in-ports of P_j to which P_i connects). Intuitively, π_i^j denotes the guarantees the ith protocol gives to the jth one. Note that we do not consider a predicate π_i^{H}. This simplifies our presentation and is without loss of generality since we are only interested in the compositionality properties of the composed protocol. We denote by

$$\pi_i = \bigwedge_{j \neq i} \pi_i^j. \tag{2}$$

the guarantees the ith protocol provides to other protocols. Note that π_i is a predicate over $\bigcup_{j \neq i}(S_i^{C,out} \cap S_j^{in})$.

In order for the composition theorems to hold, we clearly need that

$$\pi_j^i \subseteq \tau_i^j, \tag{3}$$

i.e., the guarantees τ_i^j the ith protocol expects from the jth one are actually met by the guarantees π_j^i the jth protocol offers to the ith protocol.

Obviously, in the setting above the guarantees among the protocols may be cyclic: the ith protocol provides guarantee π_i^j (and hence, τ_j^i) to the jth protocol only if the jth protocol guarantees τ_i^j, and vice versa, i.e., the jth protocol provides guarantee π_j^i (and hence, τ_i^j) to the ith protocol only if the ith protocol guarantees τ_j^i. Hence, in case $\tau_j^i \neq \texttt{true}$ *and* $\tau_i^j \neq \texttt{true}$ the dependencies between the ith and jth protocol are cyclic. The following is a concrete example.

Example 1. Say that an encryption system P_1 guarantees that the secret key is not output in plain as long as this secret key is not submitted as part of a plaintext for encryption. However, a higher-level protocol P_2 that uses that encryption system might want to encrypt plaintexts multiple times, possibly tagged with some syntactic type information. In particular, as long as the secret key in plain is not part of the plaintext of any ciphertext, this secret key will not be submitted for encryption. In other words, there is a mutual dependency between P_1 and P_2. (Obviously, in this particular case secure composition *is* possible.)

More generally, cyclic dependencies are defined as follows: Let the (directed) dependency graph $G = (V, E)$ be given by

$$V = \{V_1, \dots, V_n\}, \quad E = \{(V_i, V_j) \,:\, \tau_i^j \neq \texttt{true}\}. \tag{4}$$

If G is acyclic, we say that the dependencies between the protocols are *acyclic* or *non-mutual*, and otherwise, we say that they are *cyclic* or *mutual*.

In the following two subsections, we prove theorems for securely composing protocols, both in the case of acyclic and cyclic dependencies between the protocols. In these theorems we need to argue that the condition τ_i the ith protocol expects to be satisfied are in fact fulfilled when composing all protocols. In case of acyclic dependencies between the protocols, this is possible because the fulfillment of τ_i can be traced back to the conditions satisfied by other protocols or the honest users. In case of cyclic dependencies this is in general not possible because one runs into cycles. However, as we will see, if the predicates involved are safety properties, cyclic dependencies can be resolved. We note that the predicates informally stated in Example 1 are in fact safety predicates.

4.2 Composition in the Acyclic Case

In this section, we prove the following general composition theorem for the case of acyclic dependencies between the protocols.

Theorem 1. *For every $i = 1, \ldots, n$, let $P_i = (M_i, S_i)$ and $P'_i = (M'_i, S_i)$ be protocols as introduced above with $P_i \geq^{\tau_i}_{\mathsf{sec}} P'_i$, and assume that M'_i fulfills π_i under condition τ_i where π_i and τ_i are defined as above and condition (3) is satisfied. If the dependencies between the protocols are acyclic, we have, for every i, that*

$$P_1||\ldots||P_n \quad \geq^{\tau}_{\mathsf{sec}} \quad P_1||\ldots||P_{i-1}||P'_i||P_{i+1}||\ldots||P_n, \tag{5}$$

where $\tau := \bigwedge_{j=1}^{n} \tau_j^{\mathsf{H}}$. Moreover,

$$P_1||\ldots||P_n \quad \geq^{\tau}_{\mathsf{sec}} \quad P'_1||\ldots||P'_n. \tag{6}$$

□

Before we prove this theorem, we present useful corollaries of this theorem. The first corollary considers the case of two protocols and it easily follows from Theorem 1 using that $P_2 \geq_{\mathsf{sec}} P_2$.

Corollary 1 (Conditional Subroutine Composition). *Assume that $P_1 \geq^{\pi}_{\mathsf{sec}} P'_1$. Let $P_2 = (M_2, S_2)$ be a protocol such that M_2 i) connects to all ports over which π is defined and ii) fulfills π under condition τ where τ is a predicate over the service in-ports of P_2 to which P_1 does not connect. Then,*

$$P_1||P_2 \geq^{\tau}_{\mathsf{sec}} P'_1||P_2.$$

If $\tau = \mathsf{true}$, i.e., M_2 fulfills π unconditionally, we obtain

$$P_1||P_2 \geq_{\mathsf{sec}} P'_1||P_2.$$

□

Theorem 1 also allows to combine two protocols that are not connected via service ports:

Corollary 2 (Parallel Composition). *Assume that $P_1 \geq^{\pi_1}_{\mathsf{sec}} P'_1$ and $P_2 \geq^{\pi_2}_{\mathsf{sec}} P_2$ such that P_1 and P_2 are not connected via service ports. Then,*

$$P_1||P_2 \quad \geq^{\pi_1 \wedge \pi_2}_{\mathsf{sec}} \quad P'_1||P'_2.$$

□

Proof of Theorem 1. The proof relies on the following definition:

Definition 6. *Let M, τ, π be as in Definition 5. Then, M* fulfills π under enforced condition *τ if the predicate π is true with overwhelming probability when M interacts with machines that fulfill τ, i.e., for all sets $\overline{M}$ of machines that fulfill τ and such that $C := \{M, \overline{M}\}$ is closed, it holds that*

$$\mathsf{Pr}_{\boldsymbol{t} \leftarrow run_{C,k}} [\boldsymbol{t} \text{ satisfies } \pi] \text{ is overwhelming as a function in } k.$$

◇

Obviously, if M fulfills π under condition τ, then M fulfills π under enforced condition τ.

As a preparation for our proof, note that for $i = 1, \ldots, n$, both M'_i and M_i fulfill π_i under enforced condition τ_i. For M'_i, this is clear by assumption, and for M_i it follows from $M_i \geq^{\tau}_{\mathsf{sec}} M'_i$. (Assuming that it is not true for M_i, one obtains an honest user which cannot be simulated, contradicting the assumption that $M_i \geq^{\tau}_{\mathsf{sec}} M'_i$.) Now fix $i \in \{1, \ldots, n\}$ and set

$$\tilde{P}_i := P_1 || \ldots || P_n \text{ and } \tilde{P}'_i := P_1 || \ldots || P_{i-1} || P'_i || P_{i+1} || \ldots || P_n.$$

Theorem statement (5): We need to show that for every configuration $conf = (\tilde{P}_i, \mathsf{H}, \mathsf{A})$ of $\tilde{P}_i$, where H fulfills τ, there is a valid configuration $conf' = (\tilde{P}'_i, \mathsf{H}, \mathsf{A}')$ of $\tilde{P}'_i$ with the same H such that

$$view_{conf}(\mathsf{H}) \approx view_{conf'}(\mathsf{H}). \tag{7}$$

Step 1: We construct a new user H_i as a combination of H with all protocol machines M_j except for M_i. Note that H_i is polynomial-time, so in any case, $conf_i := (P_i, \mathsf{H}_i, \mathsf{A})$ is a configuration of P_i.

H_i *fulfills* τ_i*:* Note that this statement makes sense because H_i connects to all of M_i's service ports. The somewhat technical proof is postponed to the appendix (Lemma 2). In this proof we use that M_i fulfills π_i under enforced condition τ_i.

Step 2: Now, since H_i fulfills τ_i, the conditional simulatability of M_i guarantees the existence of a configuration $conf'_i := (P'_i, \mathsf{H}_i, \mathsf{A}')$ with

$$view_{conf_i}(\mathsf{H}_i) \approx view_{conf'_i}(\mathsf{H}_i).$$

In particular, this yields

$$view_{conf_i}(\mathsf{H}) \approx view_{conf'_i}(\mathsf{H}) \tag{8}$$

for the submachine H of H_i.

Step 3: Decomposing H_i into H and the machines M_j ($j \neq i$) yields a valid configuration $(\tilde{P}'_i, \mathsf{H}, \mathsf{A}')$ of protocol $\tilde{P}'_i$ such that (7) follows from (8) as desired.

Theorem statement (6): We show

$$P'_1 || \ldots || P'_{i-1} || P_i \ldots || P_n \quad \geq^{\tau}_{\mathsf{sec}} \quad P'_1 || \ldots || P'_i || P_{i+1} \ldots || P_n \tag{9}$$

for $i = 1, \ldots, n$ by repeatedly applying (5). The case $i = 1$ is directly implied by (5), and for $i > 1$, all P_j with $j < i$ can be set to P'_j. Then by transitivity, (9) implies (6), which completes the proof. ∎

4.3 Dealing with Mutual Dependencies – Composition in the Cyclic Case

In this section, we show that protocols can securely be composed even in case of cyclic dependencies given that the predicates considered are safety properties.

Theorem 2. *For every $i = 1, \ldots, n$, let $P_i = (M_i, S_i)$ and $P'_i = (M'_i, S_i)$ be protocols as introduced in Section 4.1 with $P_i \geq^{\tau_i}_{\mathsf{sec}} P'_i$, and assume that M'_i and M_i fulfill π_i under* *condition τ_i where π_i and τ_i are defined as in Section 4.1 and condition (3) is satisfied. Also, assume that all predicates τ_i^j, τ_i^{H}, and π_i^j are safety properties. Then, for all i, we have:*

$$P_1||\dots||P_n \quad \geq_{\mathsf{sec}}^{\tau} \quad P_1||\dots||P_{i-1}||P_i'||P_{i+1}||\dots||P_n, \tag{10}$$

where $\tau := \bigwedge_{j=1}^{n} \tau_j^{\mathsf{H}}$. Moreover,

$$P_1||\dots||P_n \quad \geq_{\mathsf{sec}}^{\tau} \quad P_1'||\dots||P_n'. \tag{11}$$

□

We note that in Theorem 2 the requirement that M_i fulfills π_i under condition τ_i can be dispensed with if service out-ports are scheduled locally (which in most scenarios is the case): The reason is that, as in the proof of Theorem 1, it easily follows that if M_i' fulfills π_i under condition τ_i, then M_i fulfills π_i under enforced condition τ_i. Now, it is not hard to see that if service out-ports are scheduled locally, then the notion of Definition 6 implies the one of Definition 5. Hence, M_i fulfills π_i under condition τ_i.

Proof of Theorem 2. For the proof of Theorem 2, we need some terminology. For a trace $\mathbf{t}$ and predicates τ and π such that τ and π are safety properties, we say that ***t** satisfies $\tau \to \pi$ at any time* if $\mathbf{t}'$ satisfies $\tau \to \pi$ for every prefix $\mathbf{t}'$ of $\mathbf{t}$.

Definition 7. *Let M, π, τ be as in Definition 5 such that π and τ are safety properties. Then,* M fulfills π under condition τ at any time *if the predicate $\tau \to \pi$ is satisfied at any time with overwhelming probability, no matter with which machines M interacts, i.e., for all sets $\overline{M}$ such that $C := \{M, \overline{M}\}$ is closed, it holds that*

$$\mathsf{Pr}_{\boldsymbol{t} \leftarrow run_{C,k}}\left[\boldsymbol{t} \textit{ satisfies } \tau \to \pi \textit{ at any time}\right] \textit{ is overwhelming as a function in } k. \tag{12}$$

◇

We can show that the above notion is equivalent to the one defined in Definition 5.

Lemma 1. *Let M, π, and τ be as in Definition 7, and such that M contains no master scheduler. Then we have that M fulfills π under condition τ at any time iff M fulfills π under condition τ.* □

Proof. The direction from left to right easily follows from the fact that if a trace $\mathbf{t}$ satisfies $\tau \to \pi$ at any time, then $\mathbf{t}$ satisfies $\tau \to \pi$.

To see the converse direction, let $\overline{M}$ be a set of machines such that $C = \{M, \overline{M}\}$ is closed and let the polynomial $p(k)$ bound the runtime of $\overline{M}$. (Note that $\overline{M}$ necessarily contains a master scheduler.) First, by definition, if a trace $\mathbf{t}$ of C does not satisfy $\tau \to \pi$ at any time, then there exists a prefix $\mathbf{t}'$ of $\mathbf{t}$ which does not satisfy $\tau \to \pi$, i.e., $\mathbf{t}'\lceil_{S_\tau} \in \tau$ but $\mathbf{t}'\lceil_{S_\pi} \notin \pi$. Let $\mathbf{t}'$ be of minimal length with this property.

We claim (*): The last transition of $\mathbf{t}'$ must be a transition of $\overline{M}$. This claim is easy to see. Assume that the last transition of $\mathbf{t}'$ is a transition of M. Let $\mathbf{t}''$ be obtained from $\mathbf{t}'$ by removing the last transition. We have that $\mathbf{t}'\lceil_{S_\tau} \in \tau$ and $\mathbf{t}'\lceil_{S_\pi} \notin \pi$. Since τ is a safety property it follows that $\mathbf{t}''\lceil_{S_\tau} \in \tau$. Since the last transition of $\mathbf{t}'$ does not contain ports in S_π

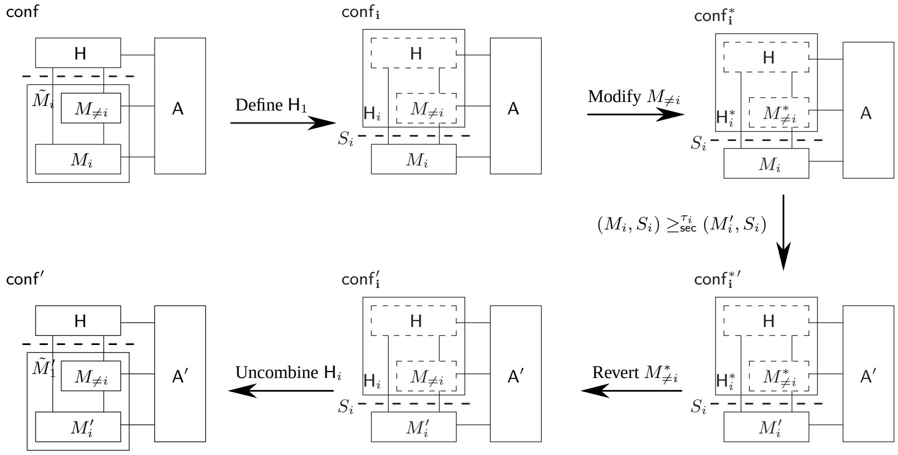

Fig. 3. Overview of the proof of Theorem 2.

(since S_π only contains in-ports of $\overline{M}$), we obtain that $\mathbf{t}'\lceil_{S_\pi} = \mathbf{t}''\lceil_{S_\pi}$. Hence, $\mathbf{t}''\lceil_{S_\pi} \notin \pi$. But this means that $\mathbf{t}''$ does not satisfy $\tau \to \pi$, in contradiction to the minimality of $\mathbf{t}'$.

Now, assume that (12) is not satisfied, i.e., $\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C,k}}[E'(k)]$, where $E'(k)$ is the event that **t** does not satisfy $\tau \to \pi$ at any time, is a non-negligible function in k. Consider the machine $\overline{M}^*$ which simulates $\overline{M}$ but at the beginning randomly chooses a position $i \in \{1, \ldots, p(k)+1\}$ and when activated for the ith time it stops (simulating $\overline{M}$). Let $C^* = \{M, \overline{M}^*\}$. We show that $\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[E(k)]$ is a non-negligible function in k, where $E(k)$ is the event "**t** does not satisfy $\tau \to \pi$". From this the lemma follows. Let "$M^*(i = j)$" denote the event that in a run of C^*, M^* picks i to be j. Then, we have that

$$\begin{aligned}
\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[E(k)] &= \sum_{j=1}^{p(k)+1} \mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[E(k) \mid M^*(i=j))] \cdot \mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[M^*(i=j)] \\
&= \frac{1}{p(k)+1} \cdot \sum_{j=1}^{p(k)+1} \mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[E(k) \mid M^*(i=j))] \\
&= \frac{1}{p(k)+1} \cdot \mathsf{Pr}_{\mathbf{t} \leftarrow run_{C,k}}[E'(k)]
\end{aligned}$$

where in the last equation we use that by (*) we have that $\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C,k}}[E'(k)] = \sum_{j=1}^{p(k)+1} \mathsf{Pr}_{\mathbf{t} \leftarrow run_{C,k}}\left[\mathbf{t} \text{ does not satisfy } \tau \to \pi \text{ and } \overline{M} \text{ performs } i \text{ transitions}\right]$. Now, since $\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C,k}}[E'(k)]$ is non-negligible, so is $\mathsf{Pr}_{\mathbf{t} \leftarrow run_{C^*,k}}[E(k)]$. ∎

We can now prove Theorem 2. For an overview of the proof, see Figure 3. We first prove (10), from which then (11) follows as in the proof of Theorem 1. Fix $i \in \{1, \ldots, n\}$ and set

$$\tilde{P}_i := P_1 || \ldots || P_n \text{ and } \tilde{P}'_i := P_1 || \ldots || P_{i-1} || P'_i || P_{i+1} || \ldots || P_n.$$

We need to show that for every configuration $\mathit{conf} = (\tilde{P}_i, \mathsf{H}, \mathsf{A})$ of $\tilde{P}_i$, where H fulfills τ, there is a valid configuration $\mathit{conf}' = (\tilde{P}'_i, \mathsf{H}, \mathsf{A}')$ of $\tilde{P}'_i$ with the same H, such that

$$\mathit{view}_{\mathit{conf}}(\mathsf{H}) \approx \mathit{view}_{\mathit{conf}'}(\mathsf{H}). \tag{13}$$

Step 1: We construct a new user H_i as a combination of H with all protocol machines M_j except for M_i. Note that H_i is polynomial-time, so in any case, $\mathit{conf}_i := (P_i, \mathsf{H}_i, \mathsf{A})$ is a configuration of P_i.

Step 2: We modify H_i into a new user H_i^* such that H_i^* fulfills τ_i. This is done by substituting all sets of submachines M_j ($j \neq i$) of H_i by sets of machines M_j^* that fulfill their respective predicates π_j *without any preconditions*. More specifically, M_j^* simulates M_j and in addition checks whether τ_j is fulfilled, i.e., whether the observed sequence of inputs on in-ports of M_j lies in τ_j. By assumption, this can be done efficiently. If τ_j is not fulfilled, then M_j^* halts immediately.

First claim regarding H_i^**:* We claim that the view of the submachine H of H_i is not changed (non-negligibly) by this modification, i.e., we claim

$$\mathit{view}_{\mathit{conf}_i}(\mathsf{H}) \approx \mathit{view}_{\mathit{conf}_i^*}(\mathsf{H}) \tag{14}$$

where $\mathit{conf}_i^* = (P_i, \mathsf{H}_i^*, \mathsf{A})$.

Assume for contradiction that (14) does not hold. Then the probability that some τ_j ($j \neq i$) is not fulfilled in a run of conf_i is non-negligible (since otherwise, conf_i and conf_i^* behave identical). Let j be such that τ_j is with non-negligible probability the *first* of all predicates τ_ℓ ($1 \leq \ell \leq n$) to become false in a run of conf_i. By "first", we mean that there is a prefix of the considered run that does not lie in τ_j, but all shorter prefixes lie in *all* τ_ℓ. (Note that by the prefix-closeness of all τ_ℓ such a prefix must exist for some j.)

Because of (1), there is thus a τ_j^r (with $r \in \{1, \dots, n, \mathsf{H}\} \setminus \{j\}$) such that with non-negligible probability, τ_j^r becomes false before any other predicate τ_ℓ, $\ell \neq j$, and $\tau_j^{r'}$, $r' \neq r$, does. As $r = \mathsf{H}$ directly contradicts the assumption on H, we may assume $r \neq \mathsf{H}$.

Now by assumption, M_r fulfills π_r, and thus, by (3) and (1), also τ_j^r under condition τ_r (in the sense of Definition 5). By Lemma 1 and the just derived statement about τ_j^r, this implies that with non-negligible probability, τ_r is false *before* τ_j is. This is a contradiction to the choice of j.

Second claim regarding H_i^**:* We claim that H_i^* fulfills τ_i (without any precondition). By (1) and the assumption on H, it suffices to prove that for any $j \neq i$, M_j^* fulfills τ_i^j without any precondition. Now since M_j fulfills π_j under condition τ_j, it also does so at any time (Lemma 1). That is, it holds with overwhelming probability that at any point during a run of M_j, π_j is true unless τ_j becomes false.

By construction, M_j^* and M_j behave identically unless τ_j becomes false. That is, also M_j^* fulfills π_j under condition τ_j at any time. In particular, by definition of M_j^*, with overwhelming probability π_j is true when M_j^* halts. It is also easy to see that π_j cannot become false after M_j^* has halted. Hence, M_j^* fulfills π_j, and thus, τ_i^j unconditionally.

Step 3: As H_i^* fulfills τ_i, the conditional simulatability of M_i guarantees the existence of a configuration $conf_i^{*\prime} := (P_i', \mathsf{H}_i^*, \mathsf{A}')$ with

$$view_{conf_i^*}(\mathsf{H}_i^*) \approx view_{conf_i^{*\prime}}(\mathsf{H}_i^*).$$

In particular, this yields

$$view_{conf_i^*}(\mathsf{H}) \approx view_{conf_i^{*\prime}}(\mathsf{H}) \tag{15}$$

for the submachine H of H_i^*.

Step 4: We substitute H_i^* again by H_i. Since, by assumption, M_i' fulfills π_i under condition τ_i, analogously to Step 2 we can show that

$$view_{conf_i^{*\prime}}(\mathsf{H}) \approx view_{conf_i'}(\mathsf{H}) \tag{16}$$

where $conf_i' = (P_i', \mathsf{H}_i, \mathsf{A}')$.

Step 5: Decomposing H_i into H and the machines M_j ($j \neq i$) yields a valid configuration $(\tilde{P}_i', \mathsf{H}, \mathsf{A}')$ of protocol $\tilde{P}_i'$ such that (13) and thus (10) follows from (14), (15) and (16) as desired. ∎

5 Applications and Examples

In this section, we provide examples substantiating the claim that conditional reactive simulatability constitutes a suitable security notion for circumventing known impossibility results of simulating interesting abstractions of cryptography. In addition, we illustrate that imposing suitable constraints on the environment may allow for a simulation proof based on much weaker assumptions on the underlying cryptography. Generally speaking, conditional reactive simulatability allows for exploiting knowledge of which protocol class will use the protocol under investigation, resulting in more fine-grained reasoning about cryptographic protocols.

More specifically, we prove that Dolev-Yao style abstractions of symmetric encryption can be correctly simulated by conditioning environments to those cases that do not cause a so-called commitment problem. For unconditional simulatability, Dolev-Yao style symmetric encryption is known not to be simulatable at all [16]. If one further constrains the environment not to create key cycles, e.g., encrypting a key with itself, we can even establish conditional simulatability based on considerably weaker assumptions on the underlying cryptographic encryption scheme. Finally, we show that conditional simulatability may naturally entail unconditional simulatability for composed protocols again.

5.1 Conditional Simulatability of Dolev-Yao Style Symmetric Encryption

For Dolev-Yao style symmetric encryption, the following so-called commitment problem inherently prevents the successful application of unconditional reactive simulatability. The ideal encryption system must somehow allow that secret keys are sent from one participant to another. This is used for example in key-exchange protocols. If the ideal system simply allows keys to be sent at any time (and typical Dolev-Yao models do allow all valid terms to be sent at any time), the following problem can occur: An honest participant first sends a ciphertext such that the adversary can see it, and later sends both the contained plaintext and the key. This behavior may even be reasonably designed into protocols, e.g., the ciphertext might be an encrypted bet that is later opened. The simulator will first learn in some abstract way that a ciphertext was sent and has to simulate it by some bitstring, which the adversary sees. Later the simulator sees abstractly that a key becomes known and that the ciphertext contains a specific application message. It cannot change the application message, thus it must simulate a key that decrypts the old ciphertext bitstring (produced without knowledge of the application message) to this specific message.

We omit a rigorous definition of the absence of the commitment problem for Dolev-Yao style symmetric encryption as given in [16, 18] but only give an informal definition for the sake of readability:

Definition 8 (No Commitment Property of Dolev-Yao Style Symmetric Encryption, informally). *The No Commitment property* NoComm *of Dolev-Yao style symmetric encryption consists of those traces of Dolev-Yao style symmetric encryption that satisfy the following trace predicate: If a term is encrypted at time* t_1 *in this trace by an honest user* u *with secret key* sk*, and at this time* sk *is not known to the adversary, then the adversary does not learn the key* sk *at any future time* t_2 *in this trace.* $\diamond$

Technically, the requirement that an adversary does not learn certain keys relies on the state of the Dolev-Yao model which keeps track of who knows which term; thus Definition 8 is syntactically not a predicate in the sense of Definition 2. However, those parts of the state that capture if an adversary already knows keys generated by honest users are uniquely determined by the preceding inputs at the service in-ports. Thus NoComm can naturally be recast as a property that is only defined at the service in-ports of the Dolev-Yao model and thus as a predicate in the sense of Definition 2 (however with a much more tedious notation).

The main result of [18] provides a simulation for those cases in which NoComm is fulfilled provided that the cryptographic encryption scheme fulfills the notion of dynamic KDM security [18]. We can now rephrase their result in our formalism to benefit from the compositionality guarantees entailed by our composition theorems. In the following, let $(\{\mathsf{TH}_{\mathcal{H}}^{\mathsf{cry\_sym,id}}\}, S_{\mathcal{H}})$ and $(\{\mathsf{M}_{\mathcal{E},u}^{\mathsf{cry\_sym,real}} \mid u \in \mathcal{H}\}, S_{\mathcal{H}})$ denote the Dolev-Yao model of symmetric encryption and its cryptographic realization from [16, 18], respectively, for a set $\mathcal{H} \subseteq \{1, \ldots, n\}$ of honest users, and an encryption scheme $\mathcal{E}$.

Theorem 3 (Conditional Reactive Simulatability of Dolev-Yao Style Symmetric Encryption). *For all symmetric encryption schemes* $\mathcal{E}$ *that satisfy dynamic KDM security [18], and for all sets* $\mathcal{H} \subseteq \{1, \ldots, n\}$ *of honest users, the realization of the Dolev-Yao model is at least as secure as the Dolev-Yao model under condition* NoComm*, i.e.,* $(\{\mathsf{M}_{\mathcal{E},u}^{\mathsf{cry\_sym,real}} \mid u \in \mathcal{H}\}, S_{\mathcal{H}}) \geq_{\mathsf{sec}}^{\mathsf{NoComm}} (\{\mathsf{TH}_{\mathcal{H}}^{\mathsf{cry\_sym,id}}\}, S_{\mathcal{H}})$. $\square$

5.2 Securely Realizing Dolev-Yao Style Symmetric Encryption with Weaker Cryptography

While Theorem 3 shows that Dolev-Yao style symmetric encryption can be conditionally simulated by excluding the commitment property, it still relies on the strong assumption that the underlying encryption scheme satisfies dynamic KDM security – a very strong, non-standard notion for which no realization in the standard model of cryptography is known. However, it turns out that this strong notion is only necessary to deal with the quite exotic case that symmetric keys are encrypted in a cyclic manner, e.g., a key with itself. Most protocols however avoid such constructions by definition, and indeed further constraining simulatability to traces that do not contain key cycles yields a simulatability result based on considerably weaker assumptions on the underlying encryption scheme. More precisely, it suffices that the encryption scheme satisfies indistinguishability under adaptive chosen-ciphertext attacks as well as integrity of ciphertexts. This is the standard security definition of authenticated symmetric encryption [49, 50], and efficient symmetric encryptions schemes provably secure in this sense exist under reasonable assumptions [51, 52].

Definition 9 (No Key Cycles for Dolev-Yao Style Symmetric Encryption, informally). *The No Key Cycles property* $\mathsf{NoKeyCycles}$ *of Dolev-Yao style symmetric encryption consists of those traces of Dolev-Yao style symmetric encryption in which honest users do not create encryptions* $E(sk_i, m_i)$ *such that* sk_{i+1} *is a subterm of* m_i *for* $i = 0, \ldots, j-1$ *for some* j*, and* sk_0 *is a subterm of* m_j*.* $\diamond$

Theorem 4 (Conditional Reactive Simulatability of Dolev-Yao Style Symmetric Encryption w/o Key Cycles). *For all authenticated symmetric encryption schemes* $\mathcal{E}$ *and all sets* $\mathcal{H} \subseteq \{1, \ldots, n\}$ *of honest users, the realization of the Dolev-Yao model is at least as secure as the Dolev-Yao model under condition* $\mathsf{NoComm} \wedge \mathsf{NoKeyCycles}$*, i.e.,* $(\{\mathsf{M}^{\mathsf{cry\_sym,real}}_{\mathcal{E},u} \mid u \in \mathcal{H}\}, S_{\mathcal{H}}) \geq^{\mathsf{NoComm} \wedge \mathsf{NoKeyCycles}}_{\mathsf{sec}} (\{\mathsf{TH}^{\mathsf{cry\_sym,id}}_{\mathcal{H}}\}, S_{\mathcal{H}})$. $\square$

5.3 Simulatable Protocols from Conditionally Simulatable Subprotocols

We finally illustrate, exploiting Corollary 1, that conditional simulatability can often be turned into unconditional simulatability again (and in fact, it seems hard to come up with a non-artificial example for which Corollary 1 does not apply). Consider a secure channel between two parties that uses Dolev-Yao style symmetric encryption as a subprimitive, which itself is only conditionally simulatable. The secure channel consists of two machines M_1 and M_2. M_1 expects a message m as input at a service port $\mathsf{in}?$, and encrypts this message with a symmetric key k shared with M_2. The encryption is computed using Dolev-Yao style symmetric encryption as a subprimitive, i.e., m is output at a service port $\mathsf{enc\_out}_1!$ and the resulting encryption e is obtained at a service port $\mathsf{enc\_in}_1?$. M_2 outputs the message at a service port $\mathsf{out}!$. We do not give a rigorous definition of this behavior here since this would presuppose introducing a significant amount of notion from [16] but it should be clear already that this secure channel neither causes a commitment problem nor any key cycles by construction. Let $(M^{\mathsf{sc}}, S^{\mathsf{sc}}) := (\{\mathsf{M}_1, \mathsf{M}_2\}, \{\mathsf{in}?, \mathsf{out}!, \mathsf{enc\_out}_1!, \mathsf{enc\_in}_1?\})$ denote the secure channel.

Theorem 5. *For all authenticated symmetric encryption schemes* $\mathcal{E}$*, and for* $\mathcal{H} = \{1, 2\}$*, the secure channel based on the realization is unconditionally at least as secure as the secure channel based on the Dolev-Yao model, i.e.,* $(M^{\mathsf{sc}}, S^{\mathsf{sc}})||(\{\mathsf{M}^{\mathsf{cry\_sym,real}}_{\mathcal{E},u} \mid u \in \mathcal{H}\}, S_{\mathcal{H}}) \geq_{\mathsf{sec}} (M^{\mathsf{sc}}, S^{\mathsf{sc}})||(\{\mathsf{TH}^{\mathsf{cry\_sym,id}}_{\mathcal{H}}\}, S_{\mathcal{H}})$. $\square$

6 Conclusion

We presented a relaxation of simulatability, one of the central concepts of modern cryptography for defining and analyzing the security of multi-party protocols, by permitting to constrain environments to adhere to certain behaviors. The resulting notion is called conditional reactive simulatability. It constitutes a more fine-grained security notion that is achievable i) for protocols for which traditional simulatability is too strong a notion, and ii) based on weaker requirements on the underlying cryptography. In addition, conditional reactive simulatability maintains the interesting property that for various protocol classes, composition of conditionally simulatable protocols yield protocols that are simulatable in the traditional sense.

We furthermore showed that despite imposing restrictions on the surrounding protocol and thus giving up the universal quantification of environments that naturally allowed for compositionality proofs in earlier works, the notion of conditional reactive simulatability still entails strong compositionality guarantees. In particular, this holds for the common case of composing so-called assume-guarantee specifications, i.e., specifications that are known to behave properly if offered suitable inputs, provided that these assumptions and guarantees constitute arbitrary trace properties that do not give rise to cyclic dependencies. We further investigated the theoretically more demanding (but arguably practically less interesting) case of cyclic dependencies among such specifications and proved a similar composition theorem under the additional assumption that conditions are expressible as safety properties.

Acknowledgments. We thank *Martín Abadi* for interesting discussions.

References

1. Michael Backes, Markus Dürmuth, Dennis Hofheinz, and Ralf Küsters. Conditional reactive simulatability. In *Computer Security, Proceedings of ESORICS 2006*, number 4189 in Lecture Notes in Computer Science, pages 424–443. Springer-Verlag, 2006.
2. Andrew Chi-Chih Yao. Theory and applications of trapdoor functions. In *23th Annual Symposium on Foundations of Computer Science, Proceedings of FOCS 1982*, pages 80–91. IEEE Computer Society, 1982.
3. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
4. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game—a completeness theorem for protocols with honest majority. In *Nineteenth Annual ACM Symposium on Theory of Computing, Proceedings of STOC 1987*, pages 218–229. ACM Press, 1987. Extended abstract.
5. Donald Beaver. Foundations of secure interactive computing. In Joan Feigenbaum, editor, *Advances in Cryptology, Proceedings of CRYPTO '91*, number 576 in Lecture Notes in Computer Science, pages 377–391. Springer-Verlag, 1992.
6. Silvio Micali and Phillip Rogaway. Secure computation. In Joan Feigenbaum, editor, *Advances in Cryptology, Proceedings of CRYPTO '91*, number 576 in Lecture Notes in Computer Science, pages 392–404. Springer-Verlag, 1992. Abstract.
7. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Secure asynchronous reactive systems. IACR ePrint Archive, March 2004.
8. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic library with nested operations. In *10th ACM Conference on Computer and Communications Security, Proceedings of CCS 2003*, pages 220–230. ACM Press, 2003. Extended abstract.
9. Michael Backes, Birgit Pfitzmann, and Michael Waidner. The reactive simulatability framework for asynchronous systems. *Information and Computation*, pages 1685–1720, 2007.
10. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42th Annual Symposium on Foundations of Computer Science, Proceedings of FOCS 2001*, pages 136–145. IEEE Computer Society, 2001.

11. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. IACR ePrint Archive, January 2005. Full and revised version of [10].
12. Birgit Pfitzmann and Michael Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *IEEE Symposium on Security and Privacy, Proceedings of SSP 2001*, pages 184–200. IEEE Computer Society, 2001.
13. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A general composition theorem for secure reactive systems. In Moni Naor, editor, *Theory of Cryptography, Proceedings of TCC 2004*, number 2951 in Lecture Notes in Computer Science, pages 336–354. Springer-Verlag, 2004.
14. R. Küsters. Simulation-Based Security with Inexhaustible Interactive Turing Machines. In *Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW-19 2006)*, pages 309–320. IEEE Computer Society, 2006.
15. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th Annual ACM Symposium on Theory of Computing, Proceedings of STOC 2002*, pages 494–503. ACM Press, 2002. Extended abstract.
16. Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable dolev-yao style cryptographic library. In *17th IEEE Computer Security Foundations Workshop, Proceedings of CSFW 2004*, pages 204–218. IEEE Computer Society, 2004.
17. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Symmetric authentication within a simulatable cryptographic library. In *Proceedings of 8th European Symposium on Research in Computer Security (ESORICS)*, volume 2808 of *Lecture Notes in Computer Science*, pages 271–290. Springer, 2003. Preprint on IACR ePrint 2003/145.
18. Michael Backes, Birgit Pfitzmann, and Andre Scedrov. Key-dependent message security under active attacks - BRSIM/UC-soundness of symbolic encryption with key cycles. In *Proceedings of 20th IEEE Computer Security Foundation Symposium (CSF)*, 2007. Preprint on IACR ePrint 2005/421.
19. Michael Backes and Birgit Pfitzmann. A cryptographically sound security proof of the Needham-Schroeder-Lowe public-key protocol. In *Proceedings of 23rd Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 2914 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2003. Preprint on IACR ePrint 2003/121.
20. Michael Backes. A cryptographically sound dolev-yao style security proof of the Otway-Rees protocol. In *Proceedings of 9th European Symposium on Research in Computer Security (ESORICS)*, volume 3193 of *Lecture Notes in Computer Science*, pages 89–108. Springer, 2004.
21. Michael Backes and Markus Duermuth. A cryptographically sound Dolev-Yao style security proof of an electronic payment system. In *Proceedings of 18th IEEE Computer Security Foundations Workshop (CSFW)*, pages 78–93, 2005.
22. Michael Backes and Birgit Pfitzmann. On the cryptographic key secrecy of the strengthened Yahalom protocol. In *Proceedings of 21st IFIP International Information Security Conference (SEC)*, pages 233–245, 2006.
23. Michael Backes, Sebastian Moedersheim, Birgit Pfitzmann, and Luca Vigano. Symbolic and cryptographic analysis of the secure WS-ReliableMessaging Scenario. In *Proceedings of Foundations of Software Science and Computational Structures (FOSSACS)*, volume 3921 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2006.
24. Michael Backes, Iliano Cervesato, Aaron D. Jaggard, Andre Scedrov, and Joe-Kai Tsay. Cryptographically sound security proofs for basic and public-key kerberos. In *Proceedings of 11th European Symposium on Research in Computer Security(ESORICS)*, volume 4189 of *Lecture Notes in Computer Science*, pages 362–383. Springer, 2006. Preprint on IACR ePrint 2006/219.
25. Michael Backes and Birgit Pfitzmann. Computational probabilistic non-interference. In *Proceedings of 7th European Symposium on Research in Computer Security (ESORICS)*, volume 2502 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2002.

26. Michael Backes and Christian Jacobi. Cryptographically sound and machine-assisted verification of security protocols. In *Proceedings of 20th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2607 of *Lecture Notes in Computer Science*, pages 675–686. Springer, 2003.
27. Michael Backes, Birgit Pfitzmann, Michael Steiner, and Michael Waidner. Polynomial fairness and liveness. In *Proceedings of 15th IEEE Computer Security Foundations Workshop (CSFW)*, pages 160–174, 2002.
28. Michael Backes and Birgit Pfitzmann. Intransitive non-interference for cryptographic purposes. In *Proceedings of 24th IEEE Symposium on Security and Privacy*, pages 140–152, 2003.
29. Michael Backes and Birgit Pfitzmann. Relating symbolic and cryptographic secrecy. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2(2):109–123, 2005.
30. Michael Backes. Quantifying probabilistic information flow in computational reactive systems. In *Proceedings of 10th European Symposium on Research in Computer Security (ESORICS)*, volume 3679 of *Lecture Notes in Computer Science*, pages 336–354. Springer, 2005.
31. Jan Camenisch and Anna Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2001*, number 2045 in Lecture Notes in Computer Science, pages 93–118. Springer-Verlag, 2001.
32. John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *Selected Areas in Cryptography, Proceedings of SAC 2002*, number 2595 in Lecture Notes in Computer Science, pages 62–75. Springer-Verlag, 2003.
33. Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology, Proceedings of CRYPTO 2001*, number 2139 in Lecture Notes in Computer Science, pages 19–40. Springer-Verlag, 2001.
34. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42th Annual Symposium on Foundations of Computer Science, Proceedings of FOCS 2001*, pages 136–145. IEEE Computer Society, 2001.
35. Yehuda Lindell, Anna Lysyanskaya, and Tal Rabin. On the composition of authenticated byzantine agreement. In *34th Annual ACM Symposium on Theory of Computing, Proceedings of STOC 2002*, pages 514–523. ACM Press, 2002.
36. Anupam Datta, Ante Derek, John C. Mitchell, Ajith Ramanathan, and Andre Scedrov. Games and the impossibility of realizable ideal functionality. In *To appear in Proceedings of Theory of Cryptography (TCC 2006)*, 2006.
37. Michael Backes, Birgit Pfitzmann, and Michael Waidner. Limits of the Reactive Simulatability/UC of Dolev-Yao models with hashes. In *Computer Security, Proceedings of ESORICS 2006*, number 4189 in Lecture Notes in Computer Science, pages 404–423. Springer-Verlag, 2006.
38. Michael Backes and Birgit Pfitzmann. Limits of the cryptographic realization of Dolev-Yao-style XOR. In *Computer Security, Proceedings of ESORICS 2005*, number 3679 in Lecture Notes in Computer Science, pages 178–196. Springer-Verlag, 2005.
39. J. Misra and M. Chandy. Proofs of networks of processes. *IEEE Transactions of Software Engineering*, 7(4):417–426, 1981.
40. C. Jones. Specification and design of (parallel) programs. In *Information Processing 83: Proceedings 9th IFIP World Congress*, pages 321–322, 1983.
41. Martín Abadi and Leslie Lamport. Conjoining specifications. *ACM Transactional on Programmming Languages and Systems.*, 17(3):507–534, 1995.
42. Dimitra Giannakopoulou, Corina S. Pasareanu, and Jamieson M. Cobleigh. Assume-guarantee verification of source code with design-level assumptions. In *Proceedings 26th International Conference on Software Engineering*, pages 211–220, 2004.
43. Heather Hinton. Composing partially-specified systems. In *IEEE Symposium on Security and Privacy, Proceedings of SSP 1998*, pages 27–39. IEEE Computer Society, 1998.

44. Dennis Hofheinz and Jörn Müller-Quade. Universally composable commitments using random oracles. In Moni Naor, editor, *Theory of Cryptography, Proceedings of TCC 2004*, number 2951 in Lecture Notes in Computer Science, pages 58–76. Springer-Verlag, 2004.
45. Manoj Prabhakaran and Amit Sahai. New notions of security: Achieving universal composability without trusted setup. In *36th Annual ACM Symposium on Theory of Computing, Proceedings of STOC 2004*, pages 242–251. ACM Press, 2004.
46. Boaz Barak and Amit Sahai. How to play almost any mental game over the net — concurrent composition via super-polynomial simulation. In *46th Annual Symposium on Foundations of Computer Science, Proceedings of FOCS 2005*, pages 543–552. IEEE Computer Society, 2005.
47. Anupam Datta, Ralf Küsters, John C. Mitchell, and Ajith Ramanathan. On the relationships between notions of simulation-based security. In *In Theory of Cryptography, Proceedings of TCC 2005*, pages 476–494, 2005.
48. Yehuda Lindell. General composition and universal composability in secure multi-party computation. In *44th Annual Symposium on Foundations of Computer Science, Proceedings of FOCS 2003*, pages 394–403. IEEE Computer Society, 2003.
49. Mihir Bellare and Phillip Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient constructions. In *Advances in Cryptology - ASIACRYPT 2000*, pages 317–330, 2000.
50. Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Advances in Cryptology - ASIACRYPT 2000*, pages 531–545, 2000.
51. Virgil D. Gligor and Pompiliu Donescu. Fast encryption and authentication: Xcbc encryption and xecb authentication modes. In *Proceedings 8th Fast Software Encryption"*, pages 82–108, 2001.
52. Charanjit Jutla. Encryption modes with almost free message integrity. In *Advances in Crptology - EUROCRYPT 2001*, pages 529–544, 2001.

A Proof of Lemma 2

Lemma 2. *In the situation of the proof of Theorem 1, user* H_i *fulfills predicate* τ_i. □

Proof. In the situation and using the notation from the proof of Theorem 1, consider running Algorithm 1. We will prove some facts about this algorithm (when run in the situation of

Algorithm 1

1: $R \leftarrow \{1, \dots, n\}$
2: **repeat**
3: $\quad S \leftarrow \{s \in R \mid \forall\, r \in R : \tau_s^r = \mathtt{true}\}$
4: $\quad R \leftarrow R \setminus S$
5: **until** $R = \emptyset$ **or** $S = \emptyset$

the proof of Theorem 1).

First claim: First, we claim that Algorithm 1 always terminates with $R = \emptyset$. It obviously suffices to prove that $S \neq \emptyset$ in each execution of Step 3: $S = \emptyset$ after any execution of Step 3 would imply that every vertex in the graph $G_R := (V_R, E_R)$ with

$$V_R = \{V_r \mid r \in R\}, \quad E_R = \{(V_a, V_b) : \tau_a^b \neq \mathtt{true}\}.$$

has nonzero out-degree, so G_R contains a cycle. But this is a contradiction, since G_R is a subgraph of the graph G (as defined in (4)), and hence, must be acyclic by assumption.

Second claim: For any $T \subseteq \{1, \ldots, n\}$, let $\mathsf{H}_{\overline{T}}$ be the combined machine that consists of H and all machines M_t with $t \notin T$. We claim that at any point during a run of Algorithm 1, the machine $\mathsf{H}_{\overline{R}}$ fulfills the predicate

$$\pi_{\overline{R}} := \left(\bigwedge_{r \notin R} \pi_r \right) \wedge \left(\bigwedge_{j=1}^{n} \tau_j^{\mathsf{H}} \right).$$

Initially, $R = \{1, \ldots, n\}$, so $\mathsf{H}_{\overline{R}} = \mathsf{H}$ and $\pi_{\overline{R}} = \bigwedge_{j=1}^{n} \tau_j^{\mathsf{H}} = \tau$, hence the statement is initially true by assumption about H. So suppose the statement is true at the start of a "**repeat**" loop of Algorithm 1. We need to show that the statement is also true after that loop.

In other words, we may assume that $\mathsf{H}_{\overline{R}}$ fulfills $\pi_{\overline{R}}$ and need to show that combining the machines M_s ($s \in S$) with $\mathsf{H}_{\overline{R}}$ yields a machine $\mathsf{H}_{\overline{R} \setminus S}$ that fulfills $\pi_{\overline{R} \setminus S}$.

By definition of combination and property fulfillment, it suffices to show that each newly added submachine M_s ($s \in S$) fulfills π_s, so fix an $s \in S$. Since M_s fulfills π_s under enforced condition τ_s, we only need to show that in all contexts in which $\mathsf{H}_{\overline{R} \setminus S}$ is run, M_s's precondition τ_s is fulfilled with overwhelming probability. But by (1) and the definition of S, τ_s is fulfilled whenever τ_s^{H} and all τ_s^r (with $r \notin R$) are fulfilled.

Using (3), τ_s^r is implied by π_r^s and thus, using (2), also by π_r. But by assumption, $\mathsf{H}_{\overline{R}}$, and hence also $\mathsf{H}_{\overline{R} \setminus S}$ fulfills $\pi_{\overline{R}}$ and τ_s^{H}. Since s was arbitrary, this shows that $\mathsf{H}_{\overline{R} \setminus S}$ fulfills all π_s ($s \in S$) and hence $\pi_{\overline{R} \setminus S}$.

Conclusion: Using the first claims just proven, we conclude that at some point during the algorithm run, $i \in S$. For the corresponding R at that point, we also have that $\mathsf{H}_{\overline{R}}$ fulfills $\pi_{\overline{R}}$. Since $i \in S$, with the same reasoning as for the second claim in this proof, we obtain that $\mathsf{H}_{\overline{R}}$ fulfills τ_i. Consequently, also the combined machine H_i, which consists of H and all M_j ($j \neq i$) fulfills τ_i since $i \notin R$ and thus, H_i contains all machines from the combination $\mathsf{H}_{\overline{R}}$. ■